#111 Appeal Brief
Berenbaum 9-4-5-5

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Patent Application

Applicant(s): Berenbaum et al.
Case: 9-4-5-5
Serial No.: 09/538,755
Filing Date: March 30, 2000
Group: 2154
Examiner: Larry D. Donaghue

I hereby certify that this paper is being deposited on this date with the U.S. Postal Service as first class mail addressed to the Commissioner for Patents, P.O. 1450, Alexandria, VA 22313-1450

Signature: [signature] Date: March 8, 2004

Title: Method and Apparatus for Splitting Packets in a Multithreaded VLIW Processor

APPEAL BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Applicants hereby appeal the final rejection dated November 14, 2003, of claims 1 through 16 of the above-identified patent application.

REAL PARTY IN INTEREST

The present application is assigned to Agere Systems Inc., as evidenced by the statement under 37 CFR 3.73 (b) submitted on March 25, 2003. The assignee, Agere Systems Inc., is the real party in interest.

RELATED APPEALS AND INTERFERENCES

A Notice of Appeal was filed on January 8, 2004 and an Appeal Brief is being submitted simultaneously, herewith, in related United States Patent Application Number 09/538,670 entitled "Method and Apparatus for Allocating Functional Units in a Multithreaded Very Large Instruction Word Processor," each assigned to the assignee of the present invention and incorporated by reference herein.

STATUS OF CLAIMS

Claims 1 through 16 are pending in the above-identified patent application. Claims 1, 2, 6-9, and 13-16 remain rejected under 35 U.S.C. § 102(b) as being anticipated by Chung et al. (United States Patent Number 5,404,469) and claims 1-5, 8-12, and 15-16 remain rejected under 35 U.S.C. § 102(b) as being anticipated by Keckler et al. (United States Patent Number 5,574,939).

STATUS OF AMENDMENTS

There have been no amendments filed subsequent to the final rejection.

SUMMARY OF INVENTION

The present invention is directed to a method and apparatus for allocating functional units in a multithreaded very large instruction word (VLIW) processor. The present invention combines the techniques of conventional very long instruction word architectures and conventional multithreaded architectures to reduce execution time within an individual program, as well as across a workload (page 7, line 12, to page 8, line 26). The present invention utilizes instruction packet splitting to recover some efficiency lost with conventional multithreaded architectures. Instruction packet splitting allows an instruction bundle to be partially issued in one cycle, with the remainder of the bundle issued during a subsequent cycle (page 10, line 15, to page 11, line 11).

ISSUES PRESENTED FOR REVIEW

i. Whether claims 1, 2, 6-9, and 13-16 are properly rejected under 35 U.S.C. § 102(b) as being anticipated by Chung et al.; and

ii. Whether claims 1-5, 8-12, and 15-16 are properly rejected under 35 U.S.C. § 102(b) as being anticipated by Keckler et al.

GROUPING OF CLAIMS

The rejected claims stand and fall together.

ARGUMENT

Independent claims 1, 8, 15, and 16 were rejected under 35 U.S.C. § 102(b) as being anticipated by Chung et al. and under 35 U.S.C. § 102(b) as being anticipated by Keckler et al. In particular, the Examiner asserts that Chung teaches an allocator for selecting and forwarding the instructions to the functional units (col. 3, line 54, to col. 4, line 63; col. 3, lines 8-29; col. 7, lines 20-40; col. 8, lines 32-55) and that Keckler teaches an allocator for selecting and forwarding the instructions to the functional units (col. 2, lines 21-38; col. 3, line 58 to col. 4, line 8).

Applicants note that, although Chung and Keckler teach that instructions are allocated to functional units, the allocation of instructions is not done independently of the type of instruction ready for execution within each thread. Chung teaches that "the processor 100 comprises four function units FU1, FU2, FU3, FU4. Illustratively, FU1 is an arithmetic unit, FU2 is a logic unit, FU3 is a load/store unit and FU4 is a branch unit." Col. 7, lines 43-46. Thus, each functional unit is *dedicated* to executing particular types of instructions and, therefore, each functional unit can only be allocated to a thread that has an instruction ready for execution wherein the instruction type matches the capability of the functional unit. The allocation of the functional units is *dependent* on the type of instructions ready for execution within each thread.

Keckler teaches that, "when several threads are competing for a given function unit, one is granted use and the others must wait. The bottom box (FIG. 1) shows a runtime interleaving of these threads in which some operations are delayed due to function unit conflicts. For example, operations A3 and A4 are locked out during the second cycle because thread C is granted those units instead." Col. 3, lines 62-67. Thus, even though the fourth functional unit is available during cycle 3 (FIG. 1), Instruction A4 of Thread A must wait until cycle 4 to execute when the corresponding functional unit type becomes available. Independent claims 1, 8, 15, and 16, as amended, require assigning instructions from at least one of said instruction packets to a plurality of said functional units, wherein said functional units can be allocated *independently* to any thread in said multithreaded instruction stream. This feature is supported in the specification on page 6, lines 28-30, (of the substitute specification) wherein it is disclosed that "the illustrative Multithreaded VLIW processor 600 includes nine

functional units 620-1 through 620-9, which can be allocated *independently* to any thread TA-TC."

Thus, Chung et al. and Keckler et al., alone or in combination, do not disclose or suggest assigning instructions from at least one of said instruction packets to a plurality of said functional units, wherein said functional units can be allocated independently to any thread in said multithreaded instruction stream, as required by independent claims 1, 8, 15, and 16, as amended.

Conclusion

The rejections of the independent claims under section §102 in view of Chung et al. and Keckler et al., alone or in any combination, are therefore believed to be improper and should be withdrawn. The rejected dependent claims are believed allowable for at least the reasons identified above with respect to the independent claims.

The attention of the Examiner and the Appeal Board to this matter is appreciated.

Respectfully,

Kevin M. Mason

Date: March 8, 2004

Kevin M. Mason
Attorney for Applicant(s)
Reg. No. 36,597
Ryan, Mason & Lewis, LLP
1300 Post Road, Suite 205
Fairfield, CT 06824
(203) 255-6560

APPENDIX

1. A multithreaded very large instruction word processor, comprising:

a plurality of functional units for executing instructions from a multithreaded instruction stream, said instructions being grouped into instruction packets by a compiler; and

an allocator that selects instructions from said instruction stream and forwards said instructions to said plurality of functional units, said allocator assigning instructions from at least one of said instruction packets to a plurality of said functional units, wherein said functional units can be allocated independently to any thread in said multithreaded instruction stream.

2. The multithreaded very large instruction word processor of claim 1, wherein said allocator assigns as many instructions from a given instruction packet as permitted by an availability of said functional units.

3. The multithreaded very large instruction word processor of claim 1, further comprising a register for storing for execution in a later cycle an indication of those instructions from a given instruction packet that cannot be allocated to a functional unit in a given cycle.

4. The multithreaded very large instruction word processor of claim 3, wherein instruction packets in which all instructions have been issued to functional units are updated from the instruction stream of said thread.

5. The multithreaded very large instruction word processor of claim 3, wherein instruction packets with instructions indicated in said register are retained.

6. The multithreaded very large instruction word processor of claim 1, wherein said allocator can split an instruction packet provided the semantics of the instruction packet assembled by the compiler are not violated.

7. The multithreaded very large instruction word processor of claim 1, wherein said allocator can split an instruction packet provided a source register for one of the instructions in a first part of said packet is not modified by one of the instructions in a second part of said packet.

8. A method for processing instructions in a multithreaded very large instruction word processor, comprising the steps of:

executing said instructions using a plurality of functional units, wherein said instructions are grouped into instruction packets by a compiler;

assigning instructions from said instruction stream to said plurality of functional units, wherein instructions from at least one of said instruction packets are assigned to a plurality of said functional units, wherein said functional units can be allocated independently to any thread in said multithreaded instruction stream; and

forwarding said selected instructions to said corresponding functional units.

9. The method of claim 8, wherein said assigning step assigns as many instructions from a given instruction packet as permitted by an availability of said functional units.

10. The method of claim 8, further comprising the step of storing for execution in a later cycle an indication of those instructions from a given instruction packet that cannot be allocated to a functional unit in a given cycle.

11. The method of claim 10, wherein instruction packets in which all instructions have been issued to functional units are updated from the instruction stream of said thread.

12. The method of claim 10, wherein instruction packets with instructions indicated in said register are retained.

13. The method of claim 8, wherein said assigning step can split an instruction packet provided the semantics of the instruction packet assembled by the compiler are not violated.

14. The method of claim 8, wherein said assigning step can split an instruction packet provided a source register for one of the instructions in a first part of said packet is not modified by one of the instructions in a second part of said packet.

15. An article of manufacture for processing instructions from an instruction stream having a plurality of threads in a multithreaded very large instruction word processor, comprising:

a computer readable medium having computer readable program code means embodied thereon, said computer readable program code means comprising program code means for causing a computer to:

execute said instructions using a plurality of functional units, wherein said instructions are grouped into instruction packets by a compiler;

assign instructions from said instruction stream to said plurality of functional units, wherein instructions from at least one of said instruction packets are assigned to a plurality of said functional units, wherein said functional units can be allocated independently to any thread in said multithreaded instruction stream; and

forward said selected instructions to said corresponding functional units.

16. A multithreaded very large instruction word processor, comprising:

a plurality of functional units for executing instructions from a multithreaded instruction stream, said instructions being grouped into instruction packets by a compiler; and

an allocator that selects instructions from said instruction stream and forwards said instructions to said plurality of functional units, said allocator assigning instructions from at least one of said instruction packets to a plurality of said functional units, wherein said functional units can be allocated independently to any thread in said

multithreaded instruction stream, provided the semantics of said instruction packet assembled by said compiler are not violated.